From: Roger Bohn
BDA April 11, 2018

3.6.8 ***Learning points Week 2: Decision trees; general Big Data Analytic concepts

3.6.8.1 Decision trees

a. Decision tree - how to interpret the results
b. Decision tree algorithm - how it is constructed. At each stage, look at the myopic best split of the cases in the current node
- Which split of *one* variable will best separate the two categories.
- This creates two new nodes of cases.

c. Stopping rules - how far down to go?
- Other algorithms make these decisions differently, but there is always an analogy
- Overfitting is what you need to avoid

d. Final decision rule: Majority of the node (if categorical classification). Mean of the node (if continuous regression).

3.6.8.2 Key concept #1: Use a **holdout sample** to evaluate performance. Train/validate/test

a. Train, validate, test subsets. Many situations require only 2 of these
b. Random selection of training data (Using Seed = 38348) means results may change slightly from run to run.
- Or, with some algorithms including Decision trees, change quite a bit in some unimportant ways.

3.6.8.3 Confusion matrix for classification problems (discrete results)

3.6.8.4 **Key concept #2: Overfitting. Models always overfit. Holdout sample tells how badly**

a. Concept of building an overfit model, and then throwing away the least important branches to reduce or eliminate overfitting.
b. How do you know if a model is overfit? Answer: look at the validation performance versus the training sample performance. If the validation performance is notably worse, you overfit.
c. Adjust model, re-do it to reduce overfitting
d. “Test sample” tells how badly

3.6.8.5 **Concept: tuning a model for better results.**

a. Different fitting methods have different tuning parameters
b. The *first result is never good enough*
c. Role of the *Complexity* parameter: how far should the model go in search of better results?
d. For tree models, tuning includes complexity and minimum split size.

3.6.8.6 Key concept #0: Variables have physical/economic/business meanings

a. “Feature engineering”
b. Redundant variables like month and year of manufacture (vs Age)
c. Study the data definitions; look at the actual numbers
d. Useless variables like specific model name (too many)
e. Variables that should probably be transformed: number of doors; air conditioning
f. Iterate here - you won’t figure it all out until you get hands dirty with the data.
g. In BDA, it’s ok to have lots of variables at start, but don’t be wedded to keeping them. We will learn how to prune out the ones that are not important.

3.6.8.7 Key concept #3: transforming the data for better fits *and* better insight/ understanding of result. aka “Feature creation”

a. Categorical variables. They may be coded as numerical 1, 2, 3… , but it’s up to us to understand and recode.
b. Discussion on t**ransforming variables**. For example, number of doors is 2 3 4 5. Should that be treated as ordinal, (numerical), categorical, or ordered category?
- (Ordered category: 3 doors is bigger than 2 doors and less than 4 doors)

c. Binary variables are usually categories, but in R, algorithms usually work the same either way. (Interpretation is clearer if you recode them as categorical.)
d. Combining variables that are logically related. Example: air conditioning + automatic air conditioning has 3 configurations, not 4.
e. This is a BIG, BIG, topic. Transforming data is one of the keys to success in analytics, and yet it’s often not taught much/not taught well. It is an art, so it takes a lot of practice.

3.6.8.8 Concept: **cleaning** data to get rid of data errors.

3.6.8.9 **Key concept # 4: Nonlinearity.**

a. The world is not linear: different causes don’t just add together to determine a final result. Models that handle nonlinearity well are desirable.
b. Nonlinearity covers both within-variable effects, and between-variable interactions.
c. Between variable linearity means f(X1, X2) = f(X1) + f(X2)
d. Within-variable linearity means f(n X1) = n f(X1). So f(X) = b X is the only linear form
- However, Xi can be a nonlinear transform of an original variable. For example, log(X), or X^3
- But it’s up to *you*, the modeler, to decide ahead of time on this nonlinear transform. A linear algorithm cannot do it for you.

e. Can create new variables that include interactions such as $X_{15} = X_1 * X_2$^2
f. So a linear model can *only* have f(X1,X2) = a + b1X1 + b2X2
g. Dealing with nonlinearity is important. Sometimes it’s vital: see key concept #3.
- We can deal with some nonlinearity “easily,” by doing a monotonic transformation of a variable. (Example: taking logarithm)
- A big virtue of decision trees (CART) and related methods is that they do not require specifying nonlinearity ahead of time.
- If nonlinear effects exist, CART will incorporate them in its model. Automatically.
- Many other kinds of models do not have this property. Nonlinearity must be guessed at and specified in advance.

3.6.8.10 Key concept #5: Knowing causality is wonderful, but for many purposes not necessary.

a. Classify: Does this patient have disease X? yes or no

- Will this person repay the loan?
- Is this transaction fraudulent?
-

b. *Predict* the outcome. Not *explain* the outcome. Not *change* the outcome

3.6.8.11 The data mining process flow.

a. Steps
b. It's iterative - you are never completely finished, but rather just decide that further improvement is not worth the effort.
c. Actually running a data fitting algorithm is a small fraction of total time.
d. Dealing with data usually more than half of time, sometimes substantially more. Gathering, cleaning, validating, studying.
e. Why is raw data usually dirty? Answer: Cleaning is expensive. Nobody bothers until they are going to use the data. Even when they clean, they only worry about problems that will affect their own application.

3.6.9 Practical advice for BDA

3.6.9.1 The darned vocabulary is not consistent

a. Parents of BDA = statistics, computer science, databases, natural sciences, social sciences
b. Each field invented its own vocabulary for same basic ideas
c. Result: lots of synonyms
d. Result: Some ambiguity. E.g. "regression" in economics ≠ "regression" in data mining

- "Logistic regression" is an oxymoron in data mining

3.6.9.2 Keep track of what you did and the results

a. Keep a notebook. Physical or electronic. Dates and times.
b. Each variant (run) of the model: what were the variables, what were the results

- Dump setup (parameters, variables, etc.) of the same runs
- Dump results of useful runs into text files.
- Use time stamps as an easy way to keep track of variants

c. Post-processing of the results
d. Example: was this run done with or without the tax variable?
e. Annotate (put notes on or into) your printouts! E.g. Excel data files.
f. Use long file names e.g. "No tax"

3.6.9.3 Stability of results for different methods

a. CART decision trees are stable in the estimates, but not in the tree that's constructed.

3.6.9.4 Keep a list of hypotheses and things to try

a. Iterate until marginal time cost is not worth the likely insight / improvment

3.6.9.5 Benefits of "pair programming"

3.6.9 Practical advice for BDA

3.6.9.1 The darned vocabulary is not consistent

a. Parents of BDA = statistics, computer science, databases, natural sciences, social sciences
b. Each field invented its own vocabulary for same basic ideas
c. Result: lots of synonyms
d. Result: Some ambiguity. E.g. "regression" in economics ≠ "regression" in data mining

- "Logistic regression" is an oxymoron in data mining

3.6.9.2 Keep track of what you did and the results

a. Keep a notebook. Physical or electronic. Dates and times.
b. Each variant (run) of the model: what were the variables, what were the results

- Dump setup (parameters, variables, etc.) of the same runs
- Dump results of useful runs into text files.
- Use time stamps as an easy way to keep track of variants

c. Post-processing of the results
d. Example: was this run done with or without the tax variable?
e. Annotate (put notes on or into) your printouts! E.g. Excel data files.
f. Use long file names e.g. "No tax"

3.6.9.3 Stability of results for different methods

a. CART decision trees are stable in the estimates, but not in the tree that's constructed.

3.6.9.4 Keep a list of hypotheses and things to try

a. Iterate until marginal time cost is not worth the likely insight / improvment

3.6.9.5 Benefits of "pair programming"